

LT Codes Combined with Network Coding for Multihop Powerline Smart Grid Networks

Abraham Kabore, Vahid Meghdadi, Jean-Pierre Cances

Univ. Limoges, CNRS, Xlim UMR 7252, 87000 Limoges, France.

Email: {wendyida-abraham.kabore,meghdadi,cances}@xlim.fr

Abstract—This paper describes a novel approach for combining Luby Transform (LT) codes and Network Coding (NC) in the context of PowerLine Communications (PLC) smart grid networks. Multihop transmissions of LT-encoded data on PLC networks are considered and algorithms to combine data at relay nodes are proposed. Without the need to decode and then re-encode the total received data stream, the relay nodes can forward the received data stream while adding at the same time their own data. Simulation results are provided confirming the good performance of the proposed algorithms.

Index Terms—Smart grid, Narrowband PLC, Network coding (NC), Fountain codes, LT codes, Multihop communications.

I. INTRODUCTION

The goal of the smart grid is to associate to the electricity transmission and distribution networks communication technologies, in order to optimize the production, the transport and the distribution of the electrical energy. NarrowBand (NB) PLC, operating below 500 kHz, is believed to be a natural and cost effective infrastructure choice for smart grid communications [1], as reflected in the adoption of specifications like G3-PLC, PRIME, and standards like IEEE 1901.2 and ITU-T G.hnem.

The characteristics of the power lines and their mode of operation make the PLC channels non-stationary, highly attenuated and very noisy. To ensure reliable transmissions on PLC channels, the conventional strategies use a fixed rate Forward Error Correction (FEC) code associated with an Automatic Repeat reQuest (ARQ) retransmission mechanism in case the FEC code has failed. Since ARQ schemes can lead to a high number of retransmissions and acknowledgments, erasure codes like fountain codes, which exhibit near-optimal overhead, are preferred in replacement of the ARQ mechanism [2], [3], [4]. Network coding is attractive for the PLC smart grid networks with the aim of increasing their performance in terms of speed, reliability and efficiency, as demonstrated by the extensive literature on this topic [5]-[6]. For a multipoint-to-point transmission, performing network coding on several LT flows to obtain a larger LT code is not a trivial problem. Indeed, LT codes are extremely sensitive to the statistical degree properties of the encoded packets. Such properties can be altered by a “naive” network coding. The problem of distributed LT coding is addressed in this paper for the specific NB PLC smart grid architecture. The association of network coding and LT coding increases the size of the generated LT

code and, if optimally combined, the resulting LT code is better in terms of overhead.

The IEEE 34-node power distribution network described in [7] and shown in the Figure 1 reveals that the PLC smart grid communication networks presents a tree and radial topology, which is considered in this paper. In fact, the network is

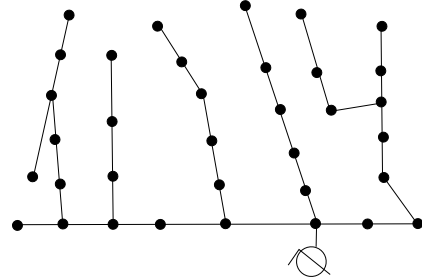


Fig. 1: Node test distribution inspired from the IEEE 34 test model.

modeled as a “line-network” gathering the information from the leaves (house meters) to the master (concentrator), as presented in the Figure 2-(a).

When polled, each source node sends its data to the concentrator either directly or through other intermediate source nodes that function as repeaters [8]. A repeating function is defined as an integral part of the MAC layer of NB-PLC smart grid networks for the purposes of range extension. This type

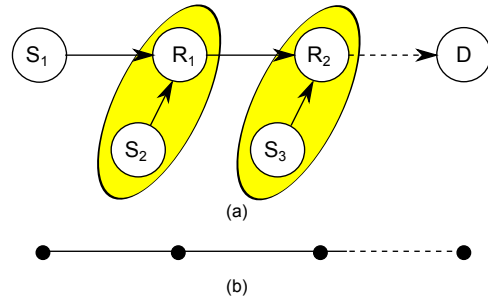


Fig. 2: (a) PLC smart grid data gathering model, (b) equivalent node model presentation as in the IEEE 34 test model

of network is a “simplified” version of the well known “Y networks” configuration, extensively addressed in the literature [9]- [10]. One major difference from the Y-networks, is that

the relay nodes are forwarding the received LT-encoded data stream from a downstream node while adding at the same time their own data i.e., all the packets of the relay nodes are of degree 1. To the best of our knowledge, this approach has barely been addressed in the literature so far.

The contributions of the paper are the following:

- The problem of distributed LT codes is analyzed for the special architecture of NB PLC smart grid networks, where a source node must relay an other source node packets while adding its own packets.
- A relaying algorithm based on the conjoint probability on the received degree from S_1 and the outgoing degree from the relay is provided. The performance evaluation of the proposed algorithm are presented.

The rest of this paper is organized as follows: Section II presents the system and gives a background on LT codes. Then in Section III, we describe the strategies and algorithms of network coding at the relays that are retained and we expose their design principles. Section IV presents the simulation results based on the above schemes confirming the system performance. Finally, Section V concludes the paper.

II. SYSTEM MODEL

Considering the system model of the Figure 1, one can notice a repeating pattern in the network. Most of the configurations encountered in the NB-PLC networks can be summarized by an elementary “pattern” consisting of one downstream source node, a bottleneck relay node and the sink node. Our set-up is outlined as follows:

- The downstream node, called S_1 , transmits an LT encoded message block of length K_1 .
- The upstream node, called S_2 , is collocated with the relay (R_1). S_2 is also transmitting a message block of length K_2 to the sink. S_2 (S_1) and upstream (downstream) node will be used interchangeably throughout this paper.
- When the sink decodes a particular source, it sends an acknowledgment to that source. The relay stop relaying the discovered packets and the corresponding source stops transmitting.
- There is no buffering for the received packets and there is no LT-decoding at the relay nodes.

In this section a short description of LT codes is given [11]. Suppose we have to transmit K packets of information. The coded packet t_n is produced from the source block M , which comprises K source messages $M = [m_1, m_2, \dots, m_K]$, by:

- Generating a random variable denoted by d_n from the predefined degree distribution.
- t_n is obtained by the bitwise sum (mod 2) of d_n packets chosen *uniformly at random* from the K packets.

At the decoder side, upon receiving a coded packet, the belief propagation LT decoder executes the following algorithm:

- If the packet is of degree 1, the packet is considered discovered. Then all the previously received and all the future packets involving this discovered packet are “xored” with the discovered packet. This is to remove its effect and to obtain lower degree packets.

- If during the process of step (i) degree 1 packets are generated, repeat the step (i).

- repeat step (i) until all the K packets are discovered.

The performance of the LT decoder is very dependent on the degree distribution from which the degrees of the coded packets are chosen. This degree distribution is called output degree distribution. The optimal output degree distribution achieving capacity was found to be the Robust Soliton Distribution (RSD) given by μ_K :

$$\mu_K(d) = \frac{\rho(d) + \tau(d)}{\left(\sum_{i=1}^K \rho(i) + \tau(i)\right)}, \quad (1)$$

$$\rho(d) = \begin{cases} 1/K, & \text{if } d = 1. \\ 1/(d \cdot (d-1)) & \text{otherwise.} \end{cases} \quad \text{and} \quad (2)$$

$$\tau(d) = \begin{cases} S/K \cdot 1/d, & \text{for } d = 1 \dots \lfloor K/S \rfloor - 1. \\ S \cdot \ln(S/\delta)/K, & \text{for } d = \lfloor K/S \rfloor. \\ 0, & \text{otherwise.} \end{cases} \quad (3)$$

Where K is the number of packets to send, d is the degree to be sent, $S = c \cdot \ln(K/\delta) \cdot \sqrt{K}$, the parameters c and δ are used to adjust the performance. In the remainder of this paper, μ_K is a vector of size $K+1$ which denotes an RSD for K source packets. The indices of μ_K start from zero with $\mu_K(0) = 0$.

For an optimal belief propagation decoding the packets must be selected at random with uniform distribution during the encoding process. This requirement produces a binomial distribution on the number of edges connected to the check nodes of the bipartite graph representing the LT code, which is called input degree distribution. Although it is easy to satisfy this condition for one source, it is quite complicated when the source nodes are distributed and network coding is done in the intermediate nodes. This important issue is considered in this paper.

For a multihop network consisting of two sources, S_1 and S_2 (S_2 being collocated with the relay) with K_1 , K_2 information packets respectively, the objective is to generate an LT code of size $K = K_1 + K_2$ preserving the input and output degree distributions. To respect the input degree distribution at the relay node, each time the relay outputs a coded packet of degree i , the probability that this packet comprises $(i-j)$ packets from S_2 is calculated as follows:

$$\frac{\binom{K_1}{j} \binom{K_2}{i-j}}{\binom{K}{i}}, \quad 1 \leq i \leq K, \quad 0 \leq j \leq i.$$

In order to quantify the impact of a non uniform sampling, some simulations have been carried out. A multipoint-to-point transmission is considered with two source nodes, $K = \{1000, 200\}$, $K_1 = K_2 = \frac{K}{2}$. The performance in terms of the decoding success rate given a specified redundancy defined as $\epsilon = N/K$ are evaluated, where N is the number of packets sent from the relay before the sink is able to decode. We assume here that all packets of S_1 are available at the relay.

At first, all the degree 2, 3 and 4 packets are selected exclusively from the source S_1 or S_2 . Therefore a degree 2, 3 or 4 packet is never coded with the packets coming from both sources. The other packets are generated using uniform sampling from the overall packets. Then, the degree 5, 6 and 7 packets are also added to this process and are sampled in the same way as the degrees 2, 3 and 4. And then, all the packets (of degree less than $\lfloor \frac{K}{2} \rfloor$) are selected either in S_1 or in S_2 . Figure 3 shows the results. The reference curve in black, corresponds to the ideal case where we have an LT of size K .

Using this method to generate the coded packets, redundant coded packets are more likely to appear. This will induce poorer performance of the LT decoding as reflected by the gap between the non uniform sampling curves and the reference curve. This gap is particularly important, when the number of degrees selected in a non uniform manner increases. From Figure 3, we note that improvements in terms of overhead can be achieved by maintaining a uniform sampling over the set of $K_1 + K_2 = K$ packets at the relay.

III. RELAYING STRATEGY

The objective at the relay is to send packets with an RSD of size K while receiving packets with an RSD of size K_1 and having at its disposal K_2 packets. Furthermore, the relay should respect the uniform and random selection of the packets resulting in the binomial input degree distribution over all the $K = K_1 + K_2$ packets.

A. Degree distribution management

Let's define the matrix $\mathbf{P}_{(K+1) \times (K_1+1)}$, with the entry $p_{i,j}$ being the joint probability that the output packet at the relay is of degree i and comprises j packets from S_1 . In an ideal case where all the packets are available at the relay the matrix \mathbf{P} is computed as:

$$\begin{aligned} p_{i,j} &= \Pr\{d = i, d_1 = j\} \\ &= \Pr\{d = i\} \Pr\{d_1 = j \mid d = i\} \\ &= \begin{cases} \mu_K(i) \frac{\binom{K_1}{j} \binom{K_2}{i-j}}{\binom{K}{i}}, & \text{for } 0 \leq j \leq i. \\ 0, & \text{for } i+1 \leq j. \end{cases} \end{aligned} \quad (4)$$

The index begins from zero for notation purposes. For example, the 5 first rows and columns of the matrix \mathbf{P} for $c = 0.05$, $\delta = 0.5$ and $K_1 = K_2 = 50$ are computed as follows:

$$\mathbf{P}_{101 \times 51} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & \dots \\ 0.016 & 0.016 & 0 & 0 & 0 & \dots \\ 0.11 & 0.22 & 0.11 & 0 & 0 & \dots \\ 0.018 & 0.058 & 0.058 & 0.018 & 0 & \dots \\ 0.005 & 0.02 & 0.03 & 0.02 & 0.005 & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \end{bmatrix} \quad (5)$$

The relay should therefore send 22% packets of degree 2, formed by “xor-ing” one packet from the downstream source and one packet from its own packets.

Since d is the output degree, marginalizing $p(d, d_1)$ with respect to d_1 yields

$$P_{\text{out}}(d = i) = \sum_{j=0}^{K_1} p_{i,j} \quad 0 \leq i \leq K.$$

This gives a vector of size $K + 1$, obtained as the sum of the columns of the matrix \mathbf{P} , that is an RSD of size K .

The marginal probability $P_{S_1}(d_1 = j)$ is a vector of size $K_1 + 1$ that can be obtained as the sum of the rows of the matrix \mathbf{P} :

$$P_{S_1}(j) = \sum_{i=0}^K p_{i,j} \quad 0 \leq j \leq K_1$$

Since the received distribution from S_1 is an RSD, it is not always possible to have at the same time a binomial input and an RS output degree distributions at the output of the relay. To better understand this, let us take a look at the matrix given in (5) for example. According to this matrix, the relay needs to output 22% packets with $d_1 = 1$ and $d = 2$. Since we do not want to decode at the relay, it is simply not possible to have this amount of degree 1 packets from S_1 because of the shape of the RSD at S_1 .

We update \mathbf{P} to \mathbf{P}_o so that the total need for the degree j packets coming from S_1 ($P_{S_1}(j)$) is less or equal to the available percentage of degree j packets actually coming from S_1 ($\mu_{K_1}(j)$). We therefore need to modify the matrix \mathbf{P} to \mathbf{P}_o , so that the sum of the columns of \mathbf{P}_o (giving P_{out}) must correspond to an RSD and the sum of the rows of \mathbf{P}_o (giving P_{S_1}) must always be possible to generate from the received S_1 output degree distribution, in other words $P_{S_1}(j) \leq \mu_{K_1}(j)$, $\forall j$.

In the following, $\mathbf{V}(i, k : n)$ is an index notation referring to the elements $\mathbf{V}(i, k), \mathbf{V}(i, k+1), \dots, \mathbf{V}(i, n)$ of the matrix \mathbf{V} .

B. Matrix \mathbf{P}_o computation

Algorithm 1: Compute \mathbf{P}_o

Input:

$\mathbf{P}(i, j) \triangleright$ The ideal joint probability of making a degree i packet at the relay with j packets coming from S_1 .

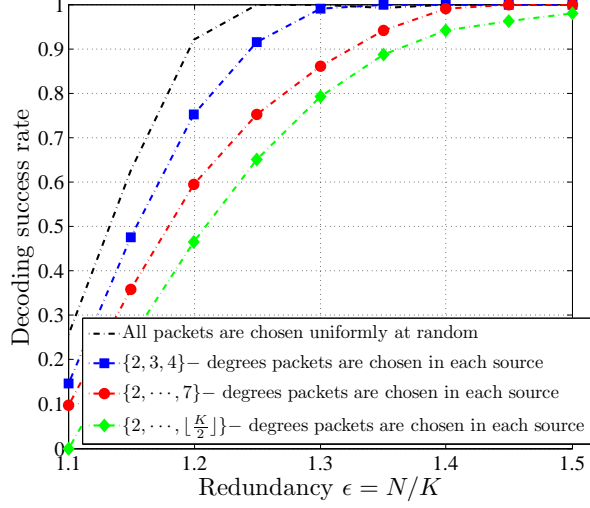
$\mu_{K_1} \triangleright$ The received output degree distribution from S_1 .

Output:

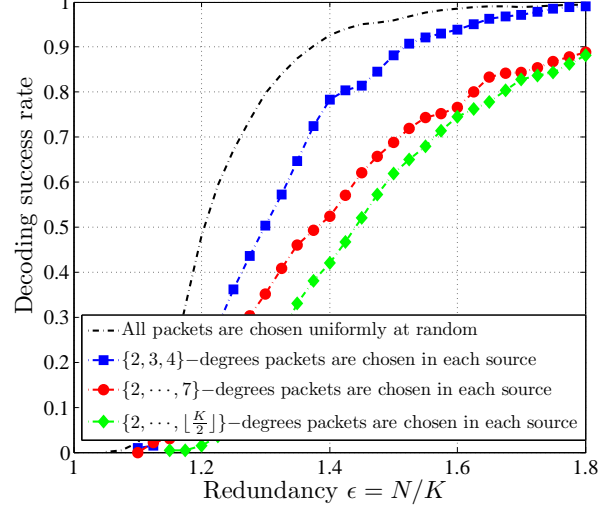
$\mathbf{P}_o(i, j) \triangleright$ A feasible joint probability of making a degree i packet at the relay with j packets coming from S_1 .

- 1 $\mathbf{P}_o = \text{zeros}(K+1, K_1+1) \triangleright$ Starts \mathbf{P}_o to a zero matrix.
 - 2 $\mu_{K_1}^{\text{residual}} = \mu_{K_1} \triangleright$ Temporary variable for μ_{K_1} .
 - 3 $\mu_{K_1}^{\text{residual}}(0) = 1 \triangleright$ Adjust the value of $\mu_{K_1}^{\text{residual}}(0)$.
 - 4 **for** $i \leftarrow 1$ **to** K **do**
 - 5 Allocate $\mu_{K_1}^{\text{residual}}(0 : i)$ to the elements in $\mathbf{P}_o(i, 0 : i)$ proportionally to the probabilities $\mathbf{P}(i, 0 : i)$ maintaining the condition :
 $\sum_{j=0}^{j=\min(K_1, i)} \mathbf{P}_o(i, j) \leq \mu_{K_1}(i)$
 - 6 Update $\mu_{K_1}^{\text{residual}}(0 : i) \triangleright$ Remove the percentage allocated for this i th step.
 - 7 **return** \mathbf{P}_o
-

To calculate \mathbf{P}_o , the elements of the vector μ_{K_1} are distributed between the different elements of the matrix \mathbf{P}_o as follows :



(a) The number of source packets $K = 1000$ and $K_1 = 500$



(b) The number of source packets $K = 200$ and $K_1 = 100$

Fig. 3: Successful decoding probability of LT codes in terms of overhead when the coded packets are not chosen uniformly at random ($c = 0.05$, $\delta = 0.5$)

- As an initialization step, the algorithm starts \mathbf{P}_o to a zero matrix. μ_{K_1} is put into a temporary variable $\mu_{K_1}^{\text{residual}}$ and $\mu_{K_1}^{\text{residual}}(0) = 1$.
- For the i th row (i starting from 1), the values of $\mathbf{P}_o(i, 0 : i)$ are increased by distributing $\mu_{K_1}^{\text{residual}}(0 : i)$ proportionally to the probabilities $\mathbf{P}(i, 0 : i)$. This allocation is done while respecting the following constraint :

$$\sum_{j=0}^{j=\min(K_1, i)} \mathbf{P}_o(i, j) \leq \mu_{K_1}(i).$$

$\mu_{K_1}^{\text{residual}}$ is then reduced, on the indexes $0, 1, \dots, i$, to remove its contribution for this i th stage.

The main steps of the matrix \mathbf{P}_o generation is summarized in algorithm 1.

We present a trace of the execution of the algorithm 1 with the following parameters $c = 0.05$, $\delta = 0.5$ and $K_1 = K_2 = 50$. The 4 first rows and columns of the matrix \mathbf{P}_o are computed in each iteration. The 4 first rows and columns of the matrix \mathbf{P} are computed as follows :

$\mathbf{P}(i, j)$	$j = 0$	$j = 1$	$j = 2$	$j = 3$...
$i = 0$	0	0	0	0	...
$i = 1$	0.0225	0.0225	0	0	...
$i = 2$	0.1099	0.2243	0.1099	0	...
$i = 3$	0.0184	0.0575	0.0575	0.0184	...
\vdots	\vdots	\vdots	\vdots	\vdots	\vdots
...

$\mathbf{P}_o(i, j)$ is initialized to a zero matrix and $\mu_{K_1}^{\text{residual}} = \mu_{K_1}$.

$\mathbf{P}_o(i, j)$	$j = 0$	$j = 1$	$j = 2$	$j = 3$...
$i = 0$	0	0	0	0	...
$i = 1$	0	0	0	0	...
$i = 2$	0	0	0	0	...
$i = 3$	0	0	0	0	...
\vdots	\vdots	\vdots	\vdots	\vdots	\vdots
...
$\mu_{K_1}^{\text{residual}}$	1	0.0316	0.4442	0.1519	...

Given that it is always possible to generate packets of degree $d_1 = 0$, we arbitrarily set

$$\mu_{K_1}^{\text{residual}}(0) = 1;$$

$\mu_{K_1}^{\text{residual}}(0)$ being the percentage of packets of degree $d_1 = 0$, coming from S_1 , employed in the merging process.

In the first iteration, we allocate $\mu_{K_1}^{\text{residual}}(0 : 1)$ to the elements in $\mathbf{P}_o(1, 0 : 1)$. The allocation is done in such a way as to best maintain the percentages found in $\mathbf{P}(1, 0 : 1)$. $\mu_{K_1}^{\text{residual}}$ is updated on the indexes $(0 : 1)$.

$\mathbf{P}_o(i, j)$	$j = 0$	$j = 1$	$j = 2$	$j = 3$...
$i = 0$	0	0	0	0	...
$i = 1$	0.0225	0.0225	0	0	...
$i = 2$	0	0	0	0	...
$i = 3$	0	0	0	0	...
\vdots	\vdots	\vdots	\vdots	\vdots	\vdots
...
$\mu_{K_1}^{\text{residual}}$	0.9775	0.091	0.4442	0.1519	...

In the second iteration, we allocate $\mu_{K_1}^{\text{residual}}(0 : 2)$ to the elements in $\mathbf{P}_o(2, 0 : 2)$. As in the previous iteration, the allocation is done in such a way as to best maintain the

percentages found in $\mathbf{P}(2, 0 : 2)$. $\mu_{K_1}^{\text{residual}}$ is updated on the indexes $(0 : 2)$ afterwards.

$\mathbf{P}_o(i, j)$	$j = 0$	$j = 1$	$j = 2$	$j = 3$...
$i = 0$	0	0	0	0	...
$i = 1$	0.016	0.016	0	0	...
$i = 2$	0.1766	0.091	0.1766	0	...
$i = 3$	0	0	0	0	...
\vdots	\vdots	\vdots	\vdots	\vdots	\vdots
...
$\mu_{K_1}^{\text{residual}}$	0.8009	0	0.2654	0.1519	...

In the third iteration, we allocate $\mu_{K_1}^{\text{residual}}(0 : 3)$ to the elements in $\mathbf{P}_o(3, 0 : 3)$ according to the same process as above. $\mu_{K_1}^{\text{residual}}$ is updated accordingly.

$\mathbf{P}_o(i, j)$	$j = 0$	$j = 1$	$j = 2$	$j = 3$...
$i = 0$	0	0	0	0	...
$i = 1$	0.016	0.016	0	0	...
$i = 2$	0.1766	0.091	0.1766	0	...
$i = 3$	0.0195	0	0.0610	0.0195	...
\vdots	\vdots	\vdots	\vdots	\vdots	\vdots
...
$\mu_{K_1}^{\text{residual}}$	0.7814	0	0.2044	0.1324	...

The update is done the same way for the i th iteration, up to $i = K$.

C. Output degree generation

Whenever a degree $d_1 = j$ is received from S_1 , the relay has two options in order to generate the output degree $d = i$.

- Either it consults the column with the index j of the matrix \mathbf{P}_o and thus forms a *pmf* (probability mass function) by normalizing this column to make it a valid *pmf*. d is drawn from this *pmf*. It then outputs a degree d packet which comprises the received packet and $d - j$ of its own source packets sampled at random.
- Or, it selects to output a coded packet including exclusively its own packets.

One notes that the relay never receives $d_1 = 0$ packets. Therefore, in order to generate the packets coming exclusively from its own set of packets, which happens with probability $P_{\text{exc-S2}} = P_{S_1}(0)$, the relay operates as follows. Upon receiving a packet from S_1 of degree $d_1 = j$ that occurs with the probability $\mu_{K_1}(j)$, at $P_{S_1}(j)/\mu_{K_1}(j)$ of the time the received packet is mixed with packets of S_2 to form the output packet as described before. And for the rest of the time, the column of indice zero of the matrix \mathbf{P}_o is used to produce packets coming exclusively from S_2 . The percentage of the packets coming exclusively from S_2 will be the same as $P_{S_1}(0)$ while the output degree distribution stays RS.

The detail of the procedure is given in the pseudo code of algorithm 2.

IV. SIMULATION RESULTS

In this section, simulation results are presented for the proposed algorithms and are essentially compared with two simulation scenarios:

Algorithm 2: Packets merging algorithm

Input:
 $\mathbf{P}_o(d, d_1)$ \triangleright The joint probability of making a degree d coded packet at the relay with d_1 packets coming from S_1 .
 Pck_{in} \triangleright The received packet from S_1 .
 μ_{K_1} \triangleright The received output degree distribution from S_1 .

Output:
 Pck_{out} \triangleright The generated packet to be sent to the sink.

- 1 $j = \text{degree}(\text{Pck}_{in})$ \triangleright The received degree.
- 2 $P_{S_1}(j) = \sum_{i=0}^K \mathbf{P}_o(i, j)$.
- 3 $P_{\text{using}}(j) = P_{S_1}(j)/\mu_{K_1}(j)$ \triangleright The probability of using Pck_{in} in the merging process.
- 4 **if** $\text{rand} \leq P_{\text{using}}(j)$ **then**
- 5 Form a *pmf*:
 $\Pr(D = d) = \mathbf{P}_o(d, j) / (\sum_{d=0}^K \mathbf{P}_o(d, j))$ for the choice of the degree d .
- 6 Choose d by sampling the *pmf*.
- 7 Sample at random $d - j$ packets from the relay to form Pck_r .
- 8 Send $\text{Pck}_{out} = \text{Pck}_{in} \oplus \text{Pck}_r$.
- 9 **else**
- 10 Choose a degree d according to the *pmf*:
 $\Pr(D = d) = \mathbf{P}_o(d, 0) / (\sum_{d=0}^K \mathbf{P}_o(d, 0))$.
- 11 Sample at random d packets from the relay to form Pck_{out} .

- a *standard LT* i.e. we consider a point-to-point transmission with only one source that transmits $(K_1 + K_2)$ LT-encoded packets to the sink.
- a time-multiplexing of two LT codes i.e. the relay alternately sends the LT-encoded packets coming from S_1 and its own LT-encoded packets.

We evaluate the performance in terms of the decoding success rate given a specified redundancy defined as $\epsilon = N/K$, where N is the number of packets sent from the relay before the sink is able to decode.

Figure 4 shows the performance of the proposed method compared with the standard LT and the time-multiplexing forwarding scheme for $K = 1000$ and $K_1 = 500$. It is observed that, the proposed merging process outperforms the time-multiplexing forwarding scheme, especially when the overhead increases. For example, for a 90% successful decoding rate, the proposed algorithm requires about 5% less overhead than the alternative time-multiplexing LT coding. For a smaller size of K ($K = 200$, $K_1 = 100$), the gap between the time-multiplexing approach and the proposed algorithm is more important, as described in Figure 5. In this Figure, one can see that the curve corresponding to the standard LT code is extremely closed to the curve corresponding to the proposed relaying algorithm. In addition, the gap between the performance of the time multiplexing and the performance

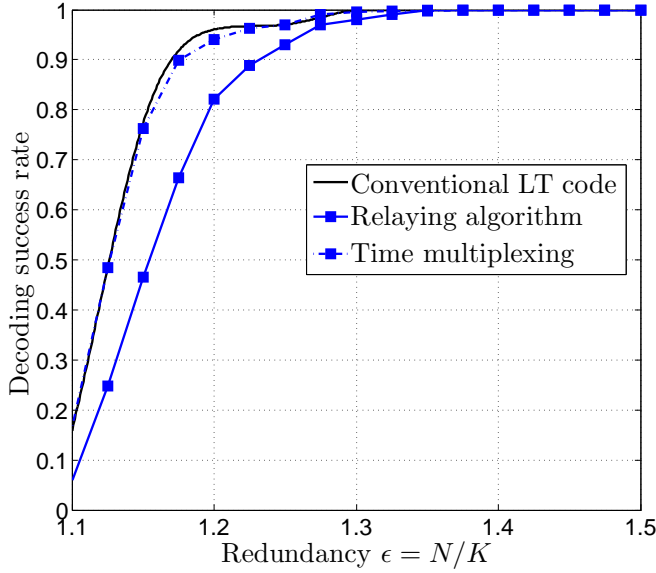


Fig. 4: Successful decoding probability of the merging process in terms of overhead (with $K_1 = K_2 = 150$, $c = 0.05$, $\delta = 0.5$)

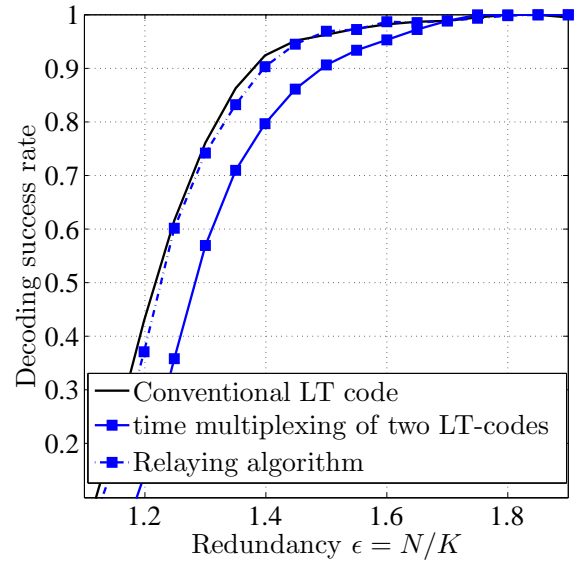


Fig. 5: Successful decoding probability of the merging process in terms of overhead (with $K_1 = K_2 = 150$, $c = 0.05$, $\delta = 0.5$)

of the proposed algorithm is wider; e.g. for 90% successful decoding percentage, the proposed algorithm requires about 10% less overhead than the alternative time-multiplexing LT coding.

V. CONCLUSION

This paper presents a strategy of relaying fountain code while using inter-session network coding for the special topology of NB-PLC networks for smart grid applications. We consider the relaying of fountain coded packets on a multihop transmission link and we propose algorithms to combine the packets at the relay, in a way to preserve the important properties that allow optimal decoding at the sink. Simulation results confirm the good performance of the proposed algorithm for a realistic network.

REFERENCES

- [1] F. Aalamifar, H. Hassanein, and G. Takahara, "Viability of powerline communication for the smart grid," in *Communications (QBSC), 2012 26th Biennial Symposium on*, pp. 19–23, May 2012.
- [2] J. W. Byers, M. Luby, M. Mitzenmacher, and A. Rege, "A digital fountain approach to reliable distribution of bulk data," in *Proceedings of the ACM SIGCOMM '98 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, SIGCOMM '98, (New York, NY, USA), pp. 56–67, ACM, 1998.
- [3] J. Byers, M. Luby, and M. Mitzenmacher, "A digital fountain approach to asynchronous reliable multicast," *Selected Areas in Communications, IEEE Journal on*, vol. 20, pp. 1528–1540, Oct 2002.
- [4] A. Kabore, V. Meghdadi, and J.-P. Cances, "Cooperative relaying in narrow-band plc networks using fountain codes," in *Power Line Communications and its Applications (ISPLC), 2014 18th IEEE International Symposium on*, pp. 306–310, March 2014.
- [5] Y. Phulpin, J. Barros, and D. Lucani, "Network coding in smart grids," in *Smart Grid Communications (SmartGridComm), 2011 IEEE International Conference on*, pp. 49–54, Oct 2011.
- [6] R. Prior, D. Lucani, Y. Phulpin, M. Nistor, and J. Barros, "Network coding protocols for smart grid communications," *Smart Grid, IEEE Transactions on*, vol. 5, pp. 1523–1531, May 2014.
- [7] D. system subcommittee, "Ieee 123 node test feeder."
- [8] J. Selga, A. Zaballos, J. Abella, and G. Corral, "Model for polling in noisy multihop systems with application to plc and amr," in *Computers and Communications, 2008. ISCC 2008. IEEE Symposium on*, pp. 664–669, July 2008.
- [9] S. Puducheri, J. Kliever, and T. Fuja, "The design and performance of distributed lt codes," *Information Theory, IEEE Transactions on*, vol. 53, pp. 3740–3754, Oct 2007.
- [10] S. Jafarizadeh, *Distributed Coding and Algorithm Optimization for Large-Scale Networked Systems*. PhD thesis, The University of SYDNEY, 2014.
- [11] M. Luby, "Lt codes," in *Foundations of Computer Science, 2002. Proceedings. The 43rd Annual IEEE Symposium on*, pp. 271–280, 2002.